Chapter Three: Contents

(Activity Generator – 31 August 2004 – LA-UR-00-1725 – TRANSIMS 3.1)

Chapter Three: Figures

Chapter Three: Tables

Chapter Three—Activity Generator

1. INTRODUCTION

1.1 Overview

The Activity Generator module generates a list of activities for each member of a synthetic population. Each activity consists of the following:

- activity type and its priority,
- starting and ending time preferences,
- a preferred mode of transportation,
- a vehicle preference (if appropriate),
- a list of possible locations for an activity, and
- a list of other participants (if the activity is shared).

The set of activities for each household is based on a household's demographics. They form the basis for determining individuals' trip plans for the region, thus resulting in travel demand sensitive to the demographics of a synthetic population. The activities also are sensitive to the network. Activity locations reflect land-use and employment data from the network. Feedback from the Route Planner or the Traffic Microsimulator provides network travel times.

1.2 Purpose

The Activity Generator has two principal purposes:

- To capture household behavior accurately—not just activity/travel patterns for individuals. Thus, if one family member takes a child to school, another need not do so.
- To ensure relative simplicity in the models for activity location. The location choice models are fairly simple. Instead of attempting to implement detailed models with network skim times etc., feedback from the Traffic Microsimulator and the Route Planner is used to refine activity location choices.

1.3 Activity Generator Major Input/Output

Fig. 1 shows how the Activity Generator uses synthetic population data, survey data, and network data to compute an activity list for every traveler.

The Activity Generator uses a synthetic population that has been located on a transportation network. At a minimum, the demographics in the population must match the demographics that are used in the Activity Generator. These demographics are used to select a suitable survey household activity pattern.

The activity survey is a representative sample of the population including travel and activity participation of all household members. Skeletal activity patterns are created by stripping locations from the survey.

The activity location table in a TRANSIMS network contains land-use and employment information that is necessary to determine the locations of activities derived from the skeletal survey patterns.

The Activity Generator's output consists of a data file that contains the activities for each traveler. This file identifies an individual and his/her corresponding household.

Each individual receives an activities list with the following attributes:

- type (e.g., home, work, shopping, school, or any other activity type collected in the survey),
- starting time range,
- ending time range,
- activity duration range,
- mode preference, and
- location.

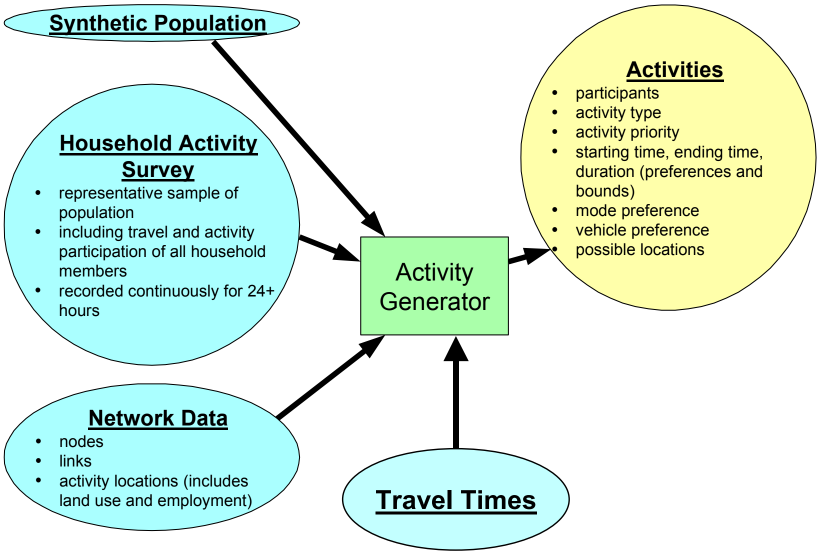
Synthetic Population
Household Activity Survey
representative sample of population
including travel and activity participation of all household members
recorded continuously for 24+ hours
Network Data
nodes
links
activity locations (includes land use and employment)
Activity Generator
Travel Times
Activities
participants
activity type
activity priority
starting time, ending time, duration (preferences and bounds)
mode preference
vehicle preference
possible locations

Fig. 1. The Activity Generator uses synthetic population, survey, and network data to compute an activity list for every traveler.

2. Module Description

2.1 Overview

TRANSIMS assumes that any two activities, separated by time and location, require travel between them. The degree of detail in both the synthetic population and activities can vary, depending on the availability of data and the study being performed. For example, a more detailed study with more realistic traffic requires a more detailed and realistic representation of the metropolitan population and the activities in which the population engages.

Each activity has a time for the activity to occur and a location. These locations must be a part of the TRANSIMS network data file. The location is one of the activity locations listed in the activity location network file. In addition to the time and location, a travel mode to reach the activity is assigned. If two or more individuals are making the same trip—in particular, a driving trip—the individuals are identified as part of the activity list.

2.2 Methodology

The Activity Generator overlays each synthetic household with a complete activity pattern. The following steps are used to create such a pattern.

Step One

- An activity survey is processed to obtain the total time spent in activity-by-activity type for each surveyed household.
- These times are weighted and summed to form a measure of total time spent in activities for each household.

Step Two

- Demographic variables of the household and the individuals in the household are selected based on which ones make the best predictors of the activity duration time.
- The predictor takes the form of a decision tree in which questions are asked at every level. The tree's terminal nodes are selected to be as homogenous as possible with respect to household activities.

Step Three

- Once a decision tree is constructed, each household from the survey is classified as belonging to one of the tree's terminal nodes.
- More than one household is usually assigned to each of the terminal nodes.

Step Four

- To allocate base activity patterns to individual households in the synthetic population, they are (1) classified according to the decision tree, and (2) given an activity pattern of one of the survey households that were similarly classified as belonging to that node.

Step Five

- Drivers and passengers on shared trips within the household are identified. The skeletal activity pattern provides all necessary information for household interactions, including shared rides.

Step Six

- Initial travel times between activity locations are estimated by using average times or calculated by using feedback from the Traffic Microsimulator or Route Planner for activity types and mode preferences.
- A modified discrete choice model based on land-use data (found in the activity location file) and travel times determines the locations of the activities, given the base activity pattern. Work locations are chosen first. Other activities are added using a multinominal logistic choice model.

2.3 Handling Freight and Itinerant Travelers

At this time, freight and itinerant travelers are handled in TRANSIMS through trip tables. The lack of data and models in these two areas makes this simplifying approximation necessary. This can be changed easily as better freight models are developed.

The following steps are taken to transform trip tables into TRANSIMS activities.

Step One Individuals (not created as part of synthetic populations) are created as vehicle drivers.

Step Two Locations for the start and end of the trips are chosen based on the zones specified in the trip tables.

Step Three One vehicle is created for each trip in the trip tables.

Note: There may be multiple trip tables representing various types of vehicles. For example, numerous trip tables could be used to represent trucks with different numbers of axles. A trip table must be available to accommodate the microsimulation of itinerant travelers.

In this version of TRANSIMS, a single vehicle is used for only one trip. If a new model generates a sequence of trips (e.g., for a single delivery truck), this would be implemented easily by creating an activity list with an activity at every stop.

The vehicle for each trip is listed in the TRANSIMS vehicle file. Each vehicle is assigned an emissions type.

3. ALGORITHM

3.1 Approach

The activity generation algorithm consists of four principal steps. Fig. 2 shows these steps graphically.

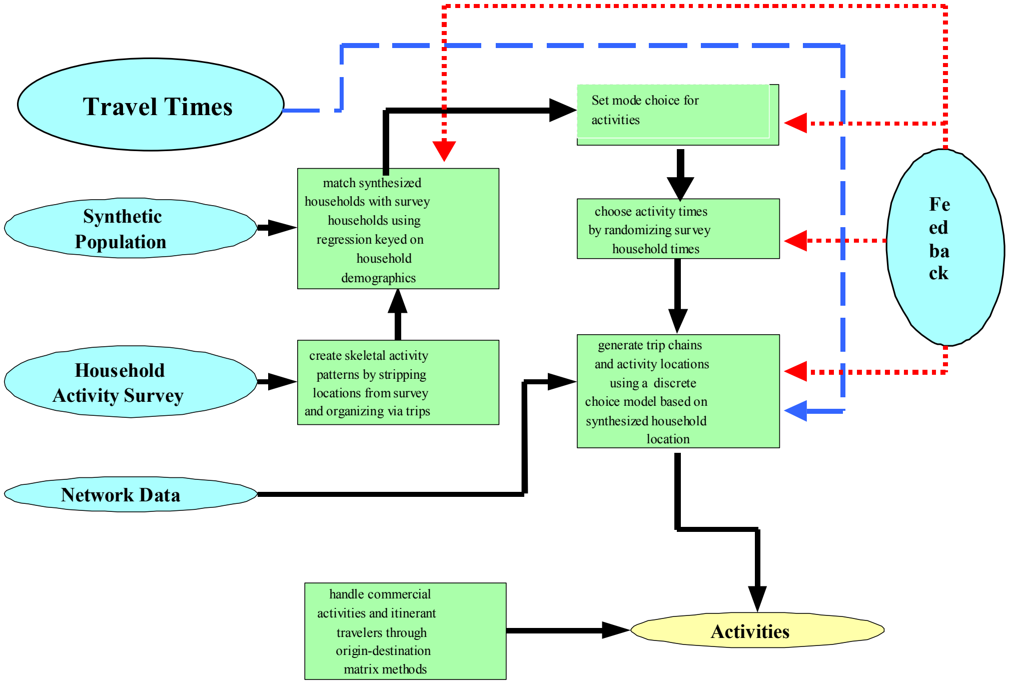
Travel Times
Synthetic Population
Household Activity Survey
Network Data
match synthesized households with survey households using regression keyed on household demographics
create skeletal activity patterns by stripping locations from survey and organizing via trips
Set mode choice for activities
choose activity times by randomizing survey household times
generate trip chains and activity locations using a discrete choice model based on synthesized household location
Feedback
handle commercial activities and itinerant travelers through origin-destination matrix methods
Activities

Fig. 2. The activity generation algorithm.

Step One

- Create skeletal patterns from the survey.
- Organize by trips the activity list for each survey household.
- Strip locations.
- The final product consists of a library of skeletal activity/travel patterns.

Step Two

- Match synthesized households with survey households.
- To make matches, use a tree-structured classification based on household demographics.
- Assign each synthetic household to a unique node in the tree.
- After the synthesized household is assigned to its tree node, select a survey household at random relative to the weights given to the survey households from the same node to obtain a matching household.
- Assign the skeletal patterns for the survey household members to the matching members in the synthesized households.

Step Three

- Group household members to trips. The final activity set includes a list of participants for each activity.

Step Four

- Because households are matched on demographics, an activity list for a person in the matching survey household is appropriate for a person in the reconstructed household except for activity location.
- Use the synthesized household location and choice models to generate new locations for the activities.

The synthetic population household demographics, survey household demographics, survey household activity/travel patterns, and a binary matching tree are inputs to the first stage, producing a skeletal activity/travel pattern for the synthesized household. Household members are matched to trips, and network land-use data are then used to generate activity locations for the skeletal pattern and create the final activity list.

3.2 Binary Tree Matching

Cross-tabulation of households by demographic variables can easily create many cells with few or no households in the survey. Instead of matching through some kind of table, household matching locates households in the terminal nodes of a binary tree. Although this tree is sensitive to the characteristics of household behavior, it is also parsimonious with respect to household characteristics that do not affect behavior.

3.3 Sample Binary Matching Tree

Fig. 3 shows a simple sample of a binary matching tree. Actual trees used in generating activities are much more complicated and will be specific to each particular transportation study.

The tree shown in Fig. 3 was constructed with an abbreviated list of three household demographic variables:

hhsize	household size
agelt5	number in household aged less than 5
age5to17	number in household aged from 5 to 17

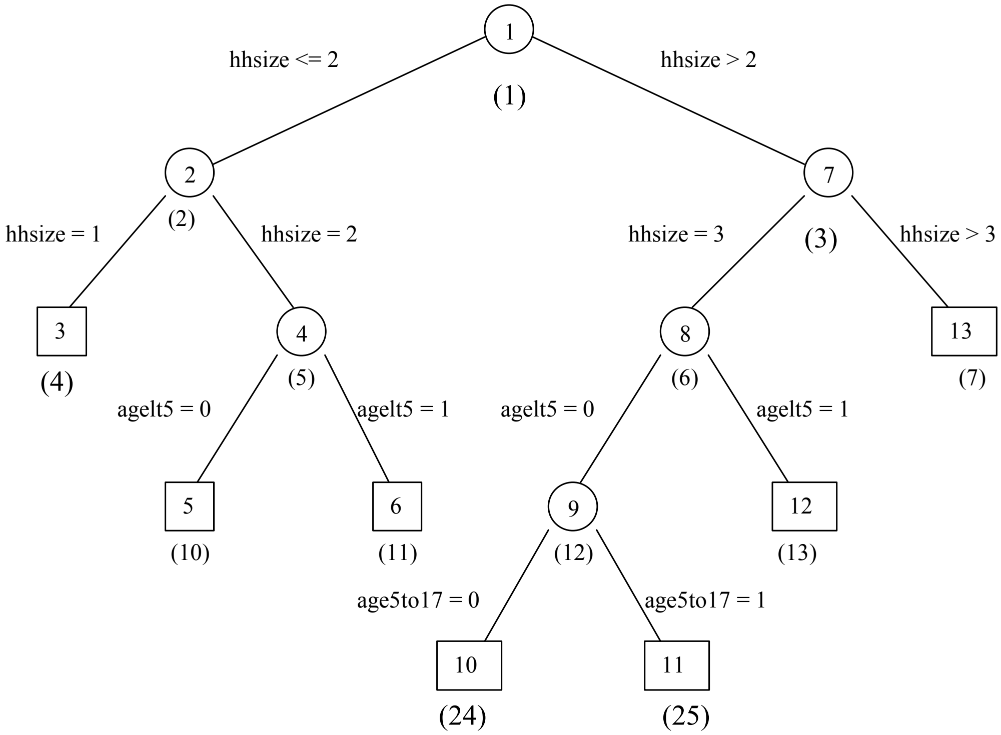

Fig. 3. This tree has 13 nodes, six of which are nonterminal nodes indicated by circles and seven are terminal nodes indicated by squares. The numbers inside the squares and circles are node numbers. The numbers beneath the nodes in parentheses are binary node numbers.

The tree in Fig. 3 has 13 nodes, including seven terminal nodes indicated by squares. Table 1 defines these seven terminal nodes.

At each non-terminal node, a household is classified further into either a left or right "child" node according to a simple rule given by a demographic variable and split point. If the appropriate variable is less than the split point, the household is classified into the left node; otherwise the right node is selected.

In the example above, the first choice (node 1) is on household size (*hhsize*). If *hhsize* is less than 2.5, the household falls somewhere in the left nodes. Otherwise, further classification proceeds to the right. The procedure continues recursively until a terminal node is reached.

Table 1. Terminal nodes.

Node	Description
3	Household size = 1
5	Household size = 2, no children less than 5
6	Household size = 2, 1 child less than 5
10	Household size = 3, no children less than 5, no children between 5 and 17
11	Household size = 3, no children less than 5, 1 child between 5 and 17
12	Household size = 3, 1 child less than 5
13	Household size greater than 3

3.4 Generating an Activities Set

The first step in generating a set of activities is to locate the synthetic household in its unique terminal node of the tree. A survey household is then selected at random from the node. For flexibility, weights are used in choosing the survey household. Each survey household has weight w_i assigned to it in the Survey Weights file. If N is the terminal node for the synthetic household, survey household i in node N is chosen with the following probability:

$$p(i) = \frac{w_i}{\sum_{j \in N} w_j}$$

3.5 Matching Individuals Within the Household

Synthetic household members are matched to members in the survey household based on the four demographic variables.

Relate Variable coded for relation to head of household
1 = spouse or partner, head of household
PUMS variable RELATE1 = 00, 01, 10
2 = child
RELATE1 = 02, 03, 06
3 = adult relative
RELATE1 = 04, 05, 07
4 = other

Work Work status
1 = full-time/part-time worker (including self employed)
2 = nonworker

Gender 1 = male
2 = female

Age Age in years

These four variables must be in the person demographics in both the synthetic population file and the survey demographic file.

3.6 Making the Best Possible Matches

Although perfect matches are not always possible, the Activity Generator does attempt to find the best matches possible. The pool of survey households in the matching node may be divided into households with and without children if the demographics at the node permit households with children. This division enables matching synthetic adults and children without resampling.

3.6.1 Rules for Children

- Children are matched to children, and adults are matched to adults.
- The children in the synthetic household are sorted by gender and age (descending). Children in the survey household are sorted in the same way, and a one-to-one match is made between the two sorted lists.
- If the survey household has more children than the synthetic household, the extra children in the survey household are ignored.

- If the synthetic household calls for more children than the survey household has, the activities of the last child in the survey household are replicated as often as necessary to create activities for the remaining children in the synthetic household.
- No survey households in the node have children; each child in the synthetic household remains at home for the entire simulation period.

3.6.2 Rules for Adults

- The adults in the synthetic household and the survey household are sorted by the following variables: `relate`, `work`, `age`, and `gender` (descending sort).
- Again, a one-to-one match is made.
- If the synthetic household contains more adults than the survey household, the activities of the last adult survey household member are replicated as often as necessary.
- At present, no further error checking is performed in the matching algorithm for adults.

3.7 Multivariate Regression Tree Program

A multivariate regression tree program can help construct a binary tree that matches synthetic households to survey households. This section provides a brief description; for a more detailed description, consult the paper by Vaughn, Speckman, and Sun[1].

A regression tree is a technique for modeling a regression relationship between a dependent variable Y and independent variables $X_1, X_2,\ldots,X_p$. Regression trees are useful when there are a large number of explanatory variables and there is an expected complex relationship between the response variable and the explanatory variables.

In these cases, tree-based methods may more easily capture non-additive behavior and thus be easier to interpret than linear models. The CART (Classification and Regression Trees) algorithm was introduced by Breiman et al.[2] and has been implemented as the tree function in the S-PLUS software package (see the paper by Clark and Pregibon[3]). The basic idea is to partition the data set into nodes defined by the predictor variables X_1, $X_2,\ldots,X_p$, and to model the response as a constant within each node. A binary tree defines these nodes.

As implemented by the CART algorithm, tree modeling consists of two stages:

- a forward recursive algorithm for "growing" the tree, and
- a second stage where the tree is "pruned back."

[1] Vaughn, K.M., Speckman, P. and Sun, D. (1999) *Identifying Relevant Socio-Demographics for Distinguishing Household Activity-Travel Patterns: A Multivariate CART Approach*.

[2] Breiman, L., J.H. Friedman, R.A. Olshen and C.J. Stone. (1984) *Classification and Regression Trees*. Chapman and Hall, London.

[3] Clark, L.A., and Pregibon, S. (1990) *Tree-based models.* In Statistical models in S. Wadsworth and Brooks/Cole, pp 377-419.

Because the growing process is only in the forward direction (once a node is defined, it cannot change), the algorithm does not necessarily produce an optimal tree. The strategy is to begin by growing a very large tree—one that probably "overfits" the data—then to use a second algorithm, thus balancing faithfulness to the data with the complexity of the tree to select a good subtree.

The philosophy is related to forward selection in the usual regression setup, where a liberal rule is adopted in the entering of variables to ensure that no important variables are omitted.

3.8 CART Algorithm

The following two sections describe how the CART algorithm grows and prunes trees.

3.8.1 Growing the Tree

To define the recursive algorithm, consider observations in a single “parent” node P_I as part of tree *T*. At the next stage, the parent node is split into two “children” nodes:

- a left node, $L_{I'}$, defined as all *I′* observations in P_I with $X_{ij} \leq t$, and
- a right node, R_I”, where $X_{ij} > t$ for a suitable choice of variable X_{ij} and cut point *t*.

The optimal variable and cut point for the split are defined in terms of the “deviance” of the node, given as

$$D(N) = \sum_{i \in N} (Y_i - \overline{Y})^2$$

the corrected total sum of squares for the observations in the node. For a potential partition split on variable X_j at cut point *t*, define the reduction in deviance from the split as

$$\Delta_{j,t} = D(P) - (D(L) + D(R))$$

A search is conducted over all *j* and *t* to find the pair *j**, *t** such that

$$\Delta_{j^*,t^*} = \max_{j,t} (\Delta_{j,t})$$

subject to,

- *I′*, *I″* > some minimum (say 10)
- D(P) > 0.01*D(total)

where *D(total)* is the deviance in the dependent variable before any splits are made. If either condition fails, the parent node is a terminal node. The algorithm recursively splits nodes until they all become terminal nodes.

3.8.2 Pruning the Tree

Prediction for a regression tree begins when the dependent variable *Y* is estimated by the mean value of *Y* in each node. However, the binary tree from the growing algorithm generally overfits the data.

Several proposals have been made to determine a better tree. One common way to assess how well a tree fits is by using it to predict a new set of data. In this case, deviance is replaced by a sum-squared prediction error. It is from this that the best subtree in the sense of minimizing prediction error can be determined. The selected tree partially depends on the data set selected to be held out. Holding out such a subset for validation may prove wasteful.

Following Breiman et al. (1984), S-PLUS implements a form of cross-validation that emulates this kind of validation without wasting data. The data set is randomly partitioned into ten approximately equal parts. Each part is held out in turn. A subtree *T*' is then re-estimated on the remaining 90% of the data, and the re-estimated tree is used to forecast the 10% held out.

Let $CV_i(T')$ denote the sum squared prediction error for the i^{th} partition. The process is repeated for all ten subsets of the data, and a total cross-validation score,

$$CV\ (T') = \sum_i CV_i(T')$$

is computed for the subtree. A subtree that minimizes (or nearly minimizes) $CV(T')$ is a good final choice for a tree that is appropriate for the data.

3.9 Multivariate Tree Algorithm

The following extended definition of deviance is used to implement the multivariate regression tree: Suppose we have dependent variables $Y_1,\ldots,Y_d$ and node *N* with *I*' observations.

Let the deviance at node *N* with respect variable Y_j be given as

$$D_j(N) = \sum_{i'} (Y_{i'j} - \overline{Y}_j)^2$$

with the total deviance at node *N*

$$D(N) = \sum_j s_j D_j(N) = \sum_j \sum_{i'} s_j (Y_{i'j} - \overline{Y}_j)^2$$

where $s_j = 1/\operatorname{var}(Y_j)$ and is a scale factor. Then for a tree T,

$$D(T) = \sum_j s_j D_j(T) = \sum_j \sum_i s_j (Y_{ij} - \overline{Y}_j)^2$$

is the scaled total sum of squares for the *I* observations in the tree. Nodes can now be split by using total deviance instead of the single-variable deviance. With this new definition of total deviance, a regression tree can be grown and pruned as before. When coupled with cross-validation, this definition of deviance can be used to prune a tree to a proper size.

In the Activity Generator, the trees are constructed using the total times households spend in 15 broadly classified activity types as the values of Y_{ij}. An additional *Y* value is the number of trips the household makes. The predictor variables are all of the demographic variables collected in the survey and the housing density. While these are reasonable variables to construct the tree, any variables could be used.

3.10 Activity Location Generation

The following steps show how the location generation algorithm works. Locations are selected by following two steps. First, a discrete choice model is used to select appropriate zones for all activities. And second, land-use variables are used to find specific activity locations within zones for each activity.

Step One Use a discrete choice model to generate all work or school locations.

Step Two Use trip-chaining discrete choice models to generate locations for other activities.

3.10.1 Work Location Model

The work location model is a simplified multinomial logistic choice model, defined with the following terms.

L Destination zone for work activity.

a(L) Attractor for work activity in zone *L*. Often expressed as *a(L) = c'x(L)*, where *x(L)* is a vector of land-use variables for zone *L*, and *c* is a coefficient vector fit by maximum likelihood. It is also possible to use other specifications for *a(L)*, including a nonparametric model for a continuous distribution fit from data. This model is described in a technical report by Speckman, Sun, and Vaughn [1998].

t(H,L) Travel time from home location *H* to work location *L*.

b_m Coefficient for mode choice *m.*

The destination zone is selected according to the probability distribution

$$p(L) = \frac{\exp(a(L) + b_m t(H,L))}{\sum_{L'} \exp(a(L') + b_m t(H,L'))}$$

Initial mode choice is taken from the survey household skeletal pattern. After the zone is selected, a specific activity location within the zone is selected at random according to weights determined by the above discrete choice model. However, in this case the actual activity locations are used in place of the distances between the zones.

$$p(A) = \frac{w_A}{\sum w_{A'}}$$

This is a simple model for several reasons. Because TRANSIMS starts with an empty network, the Activity Generator may not have access to realistic travel times. Network travel times can be fed back from the Route Planner or the Traffic Microsimulator to the Activity Generator to refine the location choice probability model. With these travel time updates, the Selector/Iteration Database module is used to develop models for mode and location choice.

3.10.2 Locations for Other Activities

To generate locations for other activities, we use a logistic multinomial choice chaining model. For example, consider a trip chain that consists of two stops on the way from work to home.

Suppose the skeletal pattern calls for travel from work location *L* to visit at L_1 by mode m_1, a second stop to shop at location L_2 by mode m_2, and finally returning home by mode m_3. The home and work locations are known. The other two destination zones, L_1 and L_2, are determined successively. For the first location, L_1, the work location *L* and the home location *H* are used in the following equation as the anchor locations of the trip

$$p(L_1) = \frac{\exp(b_{m1}t(L,L_1) + a(L_1,v) + b_{m2}t(L_1,H))}{\sum \exp(b_{m1}t(L,L_1') + a(L_1',v) + b_{m2}t(L_1',H))}$$

where the sum is over all zones. After L_1 is chosen, it will replace the work location *L* as the first anchor location for choosing the shop location L_2. In this example, separate attractors *a(L, t)* are defined for each location *L* and activity type, where the type is either *v* for "visit" or *s* for "shop." After the zone is selected, specific activity locations within the selected zone are chosen by the above formula with the activity locations replacing the zone locations.

<u>Example:</u>

> As an example, we generated an activities set for a two-adult household in Portland, Oregon. The node in the matching tree consisted of all households with two adult workers. There were 656 survey households in the node, and one was selected at random as a match. The first person had a single work activity, followed by

- a stop to visit,
- a stop to shop,
- a maintenance stop (car wash), and
- a return home.

Fig. 4 shows the set of generated activities on a map with the EMME/2 Portland network.

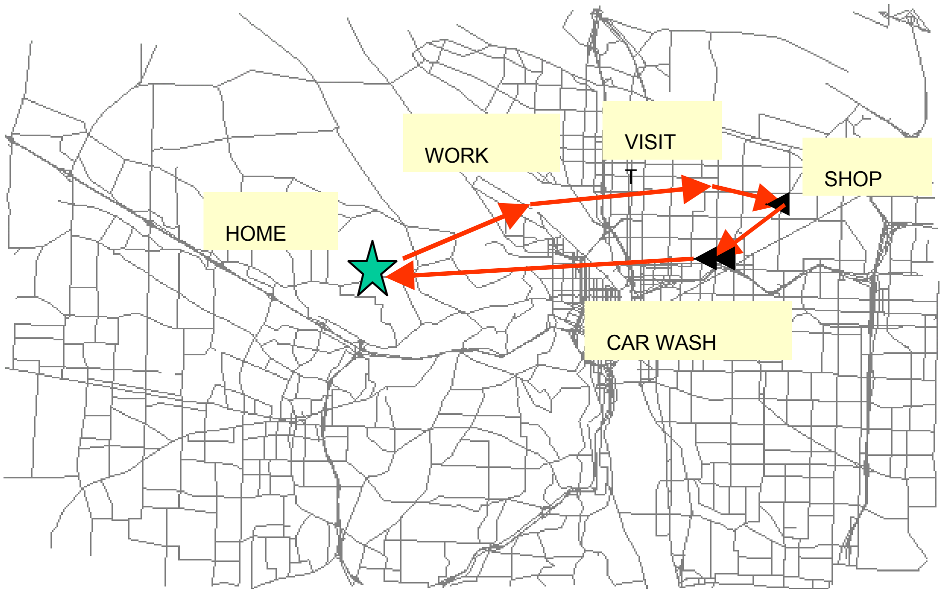

Fig. 4. Activities for the first traveler.

A second person has the following schedule:

- the top began from home to work,
- with a stop on the way,
- then went out of town on a business trip (coded “other work” in the survey),
- returned to the office, and
- finally went home.

The activity list currently has no code for out-of-town business, so as default the Activity Generator selected a second work location for the “other work” activity. The survey also had incomplete information on the last work activity with a missing location. The Activity Generator created a third location for this activity. The resulting activity pattern is shown in Fig. 5.

This example demonstrated that the Activity Generator functions well even with missing data in the survey record.

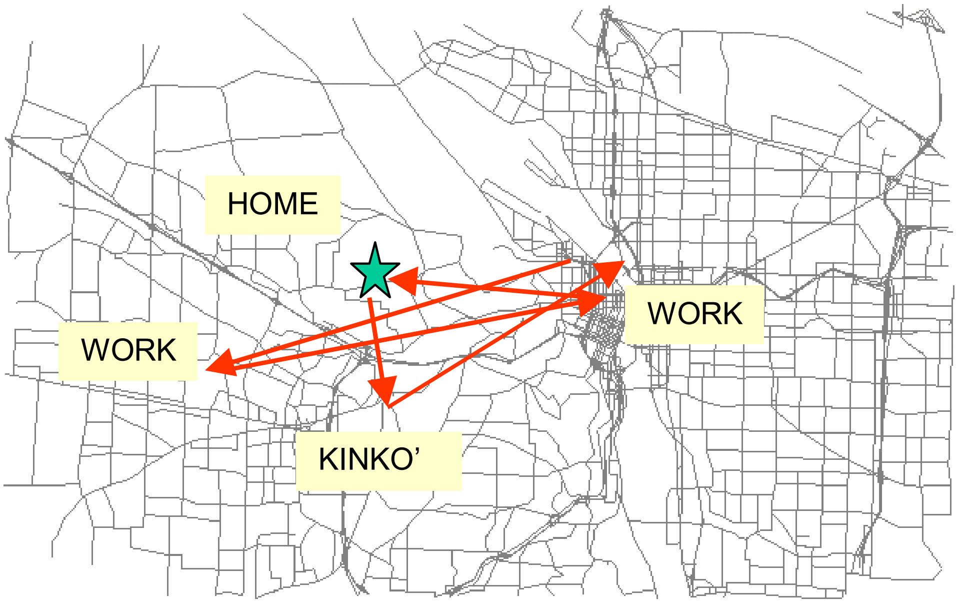

Fig. 5. Activities for the second traveler.

3.10.3 Travel Times in the Location Choice Algorithms

Zone-to-zone travel times are used in the location choice algorithms of the Activity Generator and Regenerator. There are three ways that the Activity Generator can calculate travel times. The mechanisms are specified using configuration file keys.

- Use a travel times file that contains travel times for zone pairs by mode by time of day interval. The configuration file keys `ACT_TRAVEL_TIMES_FILE` and `ACT_TRAVEL_TIME_FILE_MODES` are used to specify the file and the modes for which the file will be used.

- Code and compile a user function that will be dynamically loaded into the Activity Generator at runtime. A template function is supplied with the TRANSIMS distribution that gives the function signature and a makefile to compile the function. The user function can provide values that represent quantities other than travel times if other terms in the location choice equation are set to appropriate values. Since the return value of the user function is one of several terms inside the exponential function, care must be taken to avoid arithmetic underflow in the exponential function.

  The configuration file key `ACT_USER_FUNCTION_MODES` specifies the modes for which the function will be used. The configuration file keys `ACT_USER_FUNCTION_PARAMETERS` and `ACT_USER_FUNCTION_STRING_PARAMETERS` allow passing floating-point values and string values as parameters to the function. The configuration file key `ACT_USER_FUNCTION_SHARED_OBJECT` specifies the complete path to the user function object file.

- The default method is to compute travel times based on distance and default speeds for a given mode. Configuration file keys are used to specify the default speeds for common modes such as `auto`, `transit`, `walk`, and `bike`. The default method is used for modes that are not specified using the travel time file and travel time function configuration keys above.

Travel times are used within computation loops that are called repeatedly in the location choice algorithms; therefore, methods to calculate the return values from the user function should be computationally efficient to avoid extended run times in the Activity Generator and Regenerator. Retrieval of values from the travel times file is slow and will result in extended execution time for the Activity Generator and Regenerator.

3.11 Activity Times and Durations

Activity times are taken from skeletal activity patterns and may be changed to allow for the estimated travel time between the activities since the location of the activity will be different from the location in the survey. Travelers intent is preserved in the activity list by maintaining the duration of the activities except for the initial at-home activity.

Each activity has a preferred start time, end time, and duration. The range of each of these times is specified by a beta distribution, which requires four parameters:

- lower bound *L*,
- upper bound *U*, and
- "shape" parameters *a* and *b*.

When $a = 1$ and $b = 1$, no preference is indicated within the range *L* to *U*. If $a = -1$ and $b = -1$, the range is assumed to be 0 around the preferred time. Preferred times are taken directly from the skeletal patterns. Table 2 gives the values of the remaining parameters as currently implemented in the Activity Generator. The actual travel times between two activities given by this method may be infeasible. Using output from the Traffic Microsimulator or the Route Planner, the Selector/Iteration Database can be made to request new times for these activities.

Table 2. Settings of time parameters for activities. “Obs.” means observed time taken from skeletal pattern. The times are in hours.

Type of activity		L	U	a	b
Work	Start	Obs. - .25	Obs. + .25	1	1
	End	Obs. - .25	Obs. + .25	1	1
	Duration	Obs. - .25	Obs. + .25	1	1
Other out-of-home	Start	Obs. - .50	Obs. + .50	1	1
	End	Obs. + .50	Obs. + .50	1	1
	Duration	Obs. - .3(obs.)	Obs + .3(obs.)	1	1
AM beginning at-home	Start	0	0	-1	-1
	End	Obs. - .75	Obs. + .75	1	1
	Duration	Obs. - .75	Obs. + .75	1	1
Home during day	Start	Obs. - .75	Obs. + .75	1	1
	End	Obs. - .75	Obs. + .75	1	1
	Duration	Obs. – 1.00	Obs. + 1.00	1	1
PM end at-home	Start	Obs. - .75	Obs. + .75	1	1
	End	24.00	24.00	-1	-1
	Duration	Obs. - .75	Obs. + .75	1	1

3.12 Activity Generator Time Adjustment

Activity start and end times can be adjusted according to the following algorithm:

For each person in the household, divide the activity list into trips of the following types:

1) home to anchor
2) anchor to home
3) anchor to anchor

For each trip type, adjust the activity start/end times to account for the estimated travel time between activities as described below. Zone-to-zone travel times are computed using the means available in the Activity Generator as specified by the configuration file keys. These include travel time file, travel time function, or distance calculation.

After the individual times are adjusted for the household members, activities for drivers of shared rides are adjusted based on the passenger arrival times at departure points. The driver will wait for the latest passenger to arrive at the departure point. After the adjustment is made for the passengers, any additional wait time is cascaded forward through the remaining activities for the driver until a home activity is found with sufficient duration to absorb the wait time.

The configuration file key `ACT_ADJUST_ACTIVITY_TIMES` turns time adjustment in the Activity Generator on and off.

3.12.1 Home-to-Anchor Trips — Time Adjustment

Times are cascaded backwards from the end of the trip, the anchor activity. Times for the anchor activity are not changed. Start and end times of the intermediate activities can be changed, but the duration remains the same. The end time of the first activity in the trip that occurs at the home location may be shifted to an earlier time.

3.12.2 Anchor-to Anchor-Trips — Time Adjustment

The start and end times of the anchor activities are not changed. Time and durations of intermediate activities are adjusted to account for the estimated travel time between activities.

3.12.3 Anchor-to-Home Trips — Time Adjustment

The start and end times of the initial anchor activity in the trip are unchanged. Start and end times of the other activities in the trip may be changed, including the final home activity. The duration of the intermediate activities in the trip is maintained.

4. Parallel Computation

Parallel computation can be used in the Activity Generator to efficiently generate activities for a population. The program *MakeHouseholdFile* is used to create a set of household files that collectively contain all of the households in the population file. The household files are named according to the configuration file key ROUTER_HOUSEHOLD_FILE, and an appropriate suffix is appended. See Volume Three (*Modules*), Chapter 4 (*Route Planner*), Section 6.1 for a description of the *MakeHouseholdFile* utility. The Activity Generator can be run in parallel using the set of household files and a command line rank which is used to identify the appropriate household file to use. The configuration file key ACT_HOUSEHOLD_FILE is specified to identify the base household file name. The rank is interpreted to determine the appropriate suffix for the process. Generated activities will have the same suffix appended to the activity filename. The log file (ACT_LOG_FILE) and the problem file (ACT_PROBLEM_FILE) will also have the appended suffix.

Example:

```
% $TRANSIMS_HOME/bin/ActivityGenerator config.cfg 0
```

will start an ActivityGenerator process that looks for the household file with a suffix *.tAA*. Activities will be created for the households in this file. The activity file and log file generated from the process will have the suffix *.tAA*.

5. Activity Regeneration

5.1 Feedback to the Household Activity List

TRANSIMS is designed so that the Selector/Iteration Database module using feedback from the Route Planner and Traffic Microsimulator modules can easily request changes in activity mode preferences, times, and locations. There is a tradeoff between spending resources to make the original activity list as correct as possible and correcting the list by feedback. This is an active area of TRANSIMS research. Using feedback, it is possible to begin with a "rough" base activity list. Once the feedback begins, a refined list will emerge.

The importance of feedback to refine the activity list should not be underestimated. If feedback can improve an activity list so that extremely detailed activity models are not necessary, then national survey data and data from other sources could be used to construct the base set of activities. In this case, an extensive local survey would not be needed.

When the Route Planner or Traffic Microsimulator detects a set of impossible activities, new locations, mode preferences, and activity times can be generated or the entire household's set of activities can be regenerated. The Activity Regenerator program is used to change the existing activity list by partial regeneration of the activities or to generate a new activity list for the entire household. Feedback can also provide an updated list of travel times to be used in the activity regeneration. A file of feedback commands is used by the Activity Regenerator to specify the activity to be updated and the type of update to be applied.

5.2 Feedback Commands

The following feedback commands can be used along with the Activity Regenerator to refine a traveler's activity list. In all commands to change locations, the user has the option to supply a multiplier for the mode coefficient to be used when the new location is chosen.

5.2.1 Location Changes

- `L` – choose another location for the activity. If the activity is a shared ride, the locations of the driver/passengers are also changed. If the activity is an anchor activity as specified by the activity type, non-anchor locations in the activity list between this activity and the next anchor activity will also be changed.
- `LH` – choose new locations for all of the activities in the household except activities located at home.

- `LL` – specify a new location for an activity. The activity's location is replaced with the specified location. The location choice algorithm is not executed, and no checks are made for interdependencies on the previous location.
- `LS` – relocate a single activity. The location of the specified activity is the only change to the activity list.
- `LTR` – choose locations for activities within a tour.

5.2.2 Mode Changes

- `M` – change all activities with the specified replacement modes in a home to home tour to the new mode. If no replacement modes are specified, change all of the modes on the tour to the new mode.
- `MP` – change the specified activity to a shared ride passenger activity.
- `MS` – change the mode for a single activity. Only the specified activity's mode is changed.

5.2.3 Combination Changes

- `LM` – choose another location and change the mode of an activity. This combines the `M` command followed by the `L` command.
- `LMS` – change the mode and pick a new location for the specified activity.
- `LTYS` – change the type of the activity, then pick a new location for the specified activity.

5.2.4 Time Changes

- `T` – change the time parameters for an activity.
- `U` – adjust the household activity times based on updated travel time information.

5.2.5 Changes to the Activity List Sequence

- `AD` – add an activity to a person's activity list. The attributes of the new activity are specified as the arguments to the command.
- `RA` – remove an activity from a person's activity list.

5.2.6 Other Changes

- `AP` – change the priority assigned to an activity.

- `BSR` – break the shared rides in a household. Passengers are removed from the driver's activity list; the driver is removed from the passenger's activities, and the passengers' mode is changed to the new mode. No locations are changed for either driver or passengers.
- `R` – regenerate the household's activity list. Another activity pattern is obtained by rematching the synthetic household to a survey household in the regression tree.
- `SRL` – guarantee that locations of drivers and passengers are the same for intra-household shared ride activities in the household. The locations of the driver or passengers may be changed.
- `TP` – change the time priority assigned to an activity.
- `TY` – change the type of an activity.
- `W` – find the existing activities for the household and write them to the output file.

6. FILES

6.1 Input Files

6.1.1 Binary Tree File

A file specifies the tree structure. The variables in the tree structure are chosen as described in Section 3.3. The demographic variables used to define the tree are assumed to be the variables `<HHdata1 … HHdatan>` in the synthetic population file. The order of appearance in the synthetic population file determines the numbering of the variables in the tree—this order must match the variables in the survey population file.

These variables may come directly from census classifications or may be constructed from the survey and data records of synthetic population household members. Variables are numbered 1-n with 0 used to indicate a leaf node in the tree.

Format:

The file format has one line for each node with the following format:

```
<variable> <splitvalue> <binary node number>
```

Example:

A simple tree that assumes the following demographics for both the synthetic population and the survey population files:

```
Household demographics: hhsize alt5 a5to17
```

Variable 1 is `hhsize`, variable 2 is `alt5` (age < 5), and variable 3 is `a5to17` (ages 5 to 17). The following file gives the complete specification of the tree shown in Fig. 3:

```
1     2.5    1
1     1.5    2
0     0      4
2     0.5    5
0     0     10
0     0     11
1     3.5    3
2     0.5    6
3     0.5   12
0     0     24
0     0     25
0     0     13
0     0      7
```

6.1.2 Survey Weights File

The file containing the weights has the relative weights for the survey households in the regression tree.

Format:

```
<survey household id> <weight>
```

The weights are floating point numbers.

6.1.3 Survey Household Demographic Data File

The Activity Generator uses a set of demographic data for the survey population corresponding closely to the data for the synthetic population. The survey demographic data is contained in a TRANSIMS synthetic population file format. The tract and block group entries are set to zero, and the location is set to –1.

At a minimum, the household demographic/data variables must include the variables used for the tree-matching algorithm. The person demographics must have the variables `RELATE`, `WORK`, `GENDER`, and `AGE`.

Example:

Household Demographics: `HHSIZE ALT5 A5TO17`
Person Demographics: `RELATE WORK GENDER AGE`

The household data lines in a file with this header will have three household demographic values: (`HHSIZE`, `ALT5` and `A5TO17`). The person demographic data lines will have the four required variables: `RELATE`, `WORK`, `GENDER` and `AGE`.

6.1.4 Survey Activities File

Skeletal household activity patterns are formed from the set of activities from the survey households. The survey activity list must match the survey household demographics file. Each household and person in the demographics file must appear in the same order with activities in the activity file.

Format:

Activities are grouped sequentially for each survey household. The file consists of an ASCII file with entries separated by one or more spaces. The first line defines the column headings for the file. The columns must be in the order shown in the example below.

<u>Example:</u>

The following set of activities is for household # 200090, a household with two persons. The first person has 13 activities, including an initial at-home activity. Activities 2, 3, and 7 are out-of-home activities. The second person has two out-of-home activities.

SAMPNO	PERSNO	ACTNO	ACTID	AT_HOME	WUTHERE	MODE	DRIVER	NUNVEH	ACTSTART	ACTEND	GEOX	GEOY
200090	1	0	0	1	2	1	0	0	0	480	7651244.0000	668103.5625
200090	1	1	0	1	2	2	0	1	480	540	7651244.0000	668103.5625
200090	1	2	2	2	2	2	1	2	550	645	7655157.5000	644216.5625
200090	1	3	2	2	2	2	1	2	650	690	7654753.0000	645012.8750
200090	1	4	5	1	2	2	1	2	700	720	7651244.0000	668103.5625
200090	1	5	5	1	2	2	0	1	720	765	7651244.0000	668103.5625
200090	1	6	0	1	2	2	0	1	780	900	7651244.0000	668103.5625
200090	1	7	2	2	2	2	1	1	910	960	7655037.0000	644165.3125
200090	1	8	0	1	2	2	1	1	970	1050	7651244.0000	668103.5625
200090	1	9	4	1	2	2	0	1	1050	1095	7651244.0000	668103.5625
200090	1	10	5	1	2	2	0	1	1095	1140	7651244.0000	668103.5625
200090	1	11	0	1	2	2	0	1	1140	1303	7651244.0000	668103.5625
200090	1	12	0	1	2	2	0	1	1303	1620	7651244.0000	668103.5625
200090	2	0	0	1	2	1	0	0	0	480	7651244.0000	668103.5625
200090	2	1	0	1	2	2	0	1	480	540	7651244.0000	668103.5625
200090	2	2	2	2	2	2	2	2	550	645	7655157.5000	644216.5625
200090	2	3	2	2	2	2	2	2	650	690	7654753.0000	645012.8750
200090	2	4	5	1	2	2	2	2	700	720	7651244.0000	668103.5625
200090	2	5	5	1	2	2	0	1	720	765	7651244.0000	668103.5625
200090	2	6	0	1	2	2	0	1	780	900	7651244.0000	668103.5625
200090	2	7	5	1	2	2	0	1	900	1080	7651244.0000	668103.5625
200090	2	8	5	1	2	2	0	1	1080	1170	7651244.0000	668103.5625
200090	2	9	5	1	2	2	0	1	1170	1200	7651244.0000	668103.5625
200090	2	10	0	1	2	2	0	1	1200	1310	7651244.0000	668103.5625
200090	2	11	0	1	2	2	0	1	1310	1620	76512440000	668103.5625

Table 3. Survey Activities file format.

Field	Description	Allowed Values
`Survey Household ID`	Each survey household has a unique ID.	integer
`Person Number`	Each person in the household has a unique ID, starting with 1. Numbers are only unique within the household.	integer: 1 through household size
`Activity Number`	The activity number for each person.	integer: 0 through n: 0 = initial at-home activity of the day if necessary
`Activity Type`	Definitions may vary.	integer: 0 through n: Example: 0 = at-home activity 1 = work 2 = shop 3 = school 4 = visit 5 = other 6 = serve passenger
`At Home`	Coded 1 for activity at-home, 2 for out of home.	integer: 1 or 2
`Were-you-there`	Coded 1 if person was already at the location, 2 if not.	integer: 1 or 2
`Mode for arriving at activity`	Integer code for mode. Modes must correspond to modes in TRANSIMS mode map file used by the Route Planner and the Traffic Microsimulator.	integer: 1 through n: 1 = Walk 2 = Car 3 = Transit 4 = Light Rail 5 = Park and Ride outgoing 6 = Park and Ride incoming 7 = Bicycle 8 = Magic Move 9 = School Bus
`Driver`	Coded 1 if person was the driver, 2 if passenger. Otherwise, 0.	integer: 1, 2, or 0
`Number in Vehicle`	The number of people in vehicle	integer:1 through n
`Activity Start Time`	The time of start of activity in minutes after midnight.	integer: 0 through 2400
`Activity End Time`	The time of end of activity in minutes after midnight.	integer: 0 through 2400
`Geocode x`	The Easting geocoordinate. Must be in units agreeing with mode coefficients.	decimal
`Geocode y`	The Northing geocoordinate. Must be in units agreeing with mode coefficients.	decimal

6.1.5 Zone Data File

The zone data contain geographic data for zones used in the Activity Generator and the attractors by zone and activity type used in the location choice models. These data may be aggregated from the land-use data associated with the activity locations within the zone. These could include, for example, the total employment by SIC code or the total square footage of retail property in the zone. It should be noted that the zones could be as large as usual Traffic Analysis Zones (TAZ) or as small as the individual activity locations. The computation time however increases significantly as the number of zones increases.

Format:

There is one header line with definitions of the variables. Each variable has a one-word descriptor with no white space. The format is as follows:

```
zone NORTHING EASTING <attractor1> … <attractorN>
```

The attractor variables must correspond to the activity types in the Activity Generator. The attractors are for activity types *1 – N*.

Example:

The file below gives zone data for a test network with 10 zones. Six activity types are defined: `work`, `shop`, `school`, `visit`, `other`, and `serve_pass.`

ZONE	EASTING	NORTHING	WORK	SHOP	SCHOOL	VISIT	OTHER	SERVE_PASS
1	50	16050	2	2	-10	-10	2	1
2	50	19050	2	2	-10	-10	2	1
3	16050	35050	2	2	-10	-10	2	1
4	19050	35050	2	2	-10	-10	2	1
5	35050	19050	2	2	-10	-10	2	1
6	35050	16050	2	2	-10	-10	2	1
7	19050	50	2	2	-10	-10	2	1
8	16050	50	2	2	-10	-10	2	1
9	17500	17500	-10	-10	-10	-10	-10	1
10	17500	17500	-10	-10	3.71	3.71	–10	1

6.1.6 Mode Coefficient Data File

The coefficients b_m for distance in the choice models depend on mode *m*. This file contains coefficients by activity type by mode. These coefficients are determined by a logit fit to a simple discrete choice model given by

$$p(L) = \frac{\exp(a(L) + b_m t(H,L))}{\sum_{L'} \exp(a(L') + b_m t(H,L'))}$$

Format:

```
<coefficient> <activity type> <mode number>
```

Every activity type and mode used in the Activity Generator must have an entry in this file including the home activity type.

Example:

For activity types 1 and 2 and modes 1 - 8:

```
-0.001508     1     1
-0.001508     1     2
-0.000904     1     3
-0.001206     1     4
-0.001025     1     5
-0.001025     1     6
-0.000603     1     7
-0.001206     1     8
-0.006831     2     1
-0.006831     2     2
-0.004098     2     3
-0.005465     2     4
-0.004645     2     5
-0.004645     2     6
-0.002732     2     7
-0.005465     2     8
```

6.1.7 Feedback Commands File

The feedback command file contains commands that tell the Activity Regenerator which activities to regenerate and what action to take for each specified activity. The Household ID and Activity ID must always be specified. The command is a string that may be followed by optional command parameters. Each command is on a separate line in the file.

Format:

```
<Household Id> <Activity Id> <Command> [<command parameters>]
```

6.1.7.1 Location Changes

- `L [<mode coefficient multiplier>]` – for activities with anchor types, change location of all activities from the previous anchor activity to the next anchor activity. For non-anchor activity types, change the location only for the specified activity. For activities with anchor types, change the location of all activities and the previous anchor activity to the next anchor activity. Previous/next anchor is defined as an anchor activity type with a different location (geocode) from the specified activity. For shared or point-of-departure activities, change the location of the shared ride participants' activities. The mode coefficient in the location choice methods will be multiplied by the optional command parameter.

- `LH [mode coefficient multiplier]` – change the locations of all of the household's activities except activities located at home.

- `LL <new location>` – change the location of the specified activity to the new location. The location choice algorithm is not executed, and no checks are made for interdependencies on the current location.

- `LS [mode coefficient multiplier]` – change the location of a single activity. No other activities in the person's activity list will be changed.

- `LTR <starting activity id> [mode coefficient multiplier][time travel offset][ending activity id]` – change the locations of all activities within a tour. A tour is all activities between the specified starting activity and the end of the tour. The end of the tour is determined by the ending activity id, if specified. If the ending activity is not specified, the end of the tour is the first occurrence of the following:
  - another activity with the same location as the starting activity,
  - an activity at the home location
  - the last activity in the activity list

6.1.7.2 Mode Changes

- `M <mode value> [<replacement mode 1> ... <replacement mode n>]` – change all activities with the specified replacement modes to the new mode in the home-to home-tour that contains the specified activity. If no replacement modes are specified, all of the modes in the tour will be changed.

- `MP <mode> <vehicle ID> <driver ID> [<start time>] [<end time>]` – change the specified activity to a shared ride passenger activity.

- `MS <mode value>` -- change the mode for the activity to the integer mode value. Modes to other activities are unchanged.

6.1.7.3 Combination Changes

- `LM <mode value> [mode coefficient multiplier]` – change the mode for the activity to the mode value and then change the location for the activity. Action is identical to those of the combined `M` and `L` commands.

- `LMS <mode> [mode coefficient multiplier]` – change the mode to the specified mode, then choose a new location for the specified activity. Other activities in the activity list are not changed by this command.

- `LTYS <type> [mode coefficient multiplier]` – change the activity type to the specified type then chose a new location for the activity. Other activities in the activity list are not changed by this command.

6.1.7.4 Time Changes

- `T <start time>  [<end time>]  [<alpha parameter>] [<beta parameter>]` – change the time for the activity to start time and, if specified, end time with alpha and beta parameters on the time range.

- `U` – update the activity times for the household. Activity times will be adjusted based on the latest travel time information.

  Updated zone-to-zone travel time information can be supplied using the means available in the Activity Generator as specified by the configuration file keys. These include a travel time file or travel time function.

  Activity times are adjusted for each person in the household in the same manner as when the configuration file key `ACT_ADJUST_ACTIVITY_TIMES` is set during the initial activity generation. See section 3.12.

6.1.7.5 Changes to the Activity List Sequence

- `AD <type> <priority> <mode> <vehicle id> <time priority> <location> <starttime lower bound> <starttime upper bound> <end time lower bound> <end time upper bound> <# other participants> [<other participant 1> ... <other participant n>]` – add an activity to a person's activity list. The attributes of the new activity are specified as the arguments to the command. The vehicle ID should be -1 for non-driving modes. The number of other participants is 1 or more for shared rides. The other participant list begins with the driver's ID for both driver and passengers. If driver, then the list also contains the passengers' IDs. The driver ID is the only entry in the other participant list for passengers.

- `RA` – remove the specified activity from a person's activity list.

6.1.7.6 Other Changes

- `AP <priority>` – change the priority of the activity.

- `BSR <new passenger mode>` -- break the shared rides in a household. The passengers are removed from the driver's activity list; the driver is removed from the passenger's activities, and the passenger' mode is changed to the new mode. No locations are changed for either driver or passengers. The activity ID in this command is a placeholder since all of the shared ride activities for the household are changed.

- `R` – regenerate the entire activity list for the household by rematching with a survey household.

- `SRL` – guarantee that locations of driver and passengers are the same for shared ride activities in a household. Locations of driver or passenger are changed to achieve the location match. The activity ID must be specified and can be a dummy value since it is not used.
- `TP <priority>` – change the time priority of the activity.
- `TY <activity type>` – change the type for the activity to the specified type.
- `W` – find the existing activities for the household and write them to the partial activity file.

Example:

```
1356   7  L
1358   2  LM  3
1379  10  M   2
1380  13  MS  3
1386   4  T   420  1040  0.5  1.0
1395   R
```

In the example above,

- Activity 7 in Household 1356 will have a new location generated.
- Activity 2 in Household 1358 will have the mode changed to 3 and then a new location generated.
- Activity 10 in Household 1379 will have the mode changed to 2 and all activities on this home-to-home tour may be changed to be consistent with the new mode.
- Activity 13 in Household 1380 will have the mode changed to 3.
- Activity 4 in Household 1386 will have the start time changed to 420, the end time changed to 1040, the alpha parameter set to 0.5, and the beta parameter set to 1.0.
- Household 1395 will have all of the activities regenerated.

6.1.8 Travel Times File

The travel times file contains information about zone-to-zone travel times by mode by time of day intervals. It provides a mechanism to update the travel time information used by the location choice algorithms in the Activity Generator and Regenerator. The travel times in the file are specified by mode and a time of day interval. See Volume Three (*Modules*), Chapter 3 (*Activity Generator*), Section 9.4 for a description of how time of day intervals are specified using a time intervals file.

Each entry in the travel times file also has information about the number of data points averaged to get the travel time value and the variance of the data points. The Activity Generator and Regenerator do not use the last two fields, so default values of 0 can be entered.

Format:

```
<zone1> <zone2> <mode> <interval> <travel time> <#data points> <variance>
```

where

- `zone 1` = the number of the zone where the trip originates.
- `zone 2` = the number of the zone where the trip terminates.

- `mode` = the TRANSIMS mode number. This number must be the same as the modes used in the Activity Generator and Route Planner.

- `interval` = the number of the time of day interval for which the travel time applies. The time intervals and time of day ranges for the intervals are specified in a time intervals file.

- `travel time` = the travel time in seconds for trips from zone 1 to zone 2.

- `number of data points` = the number of data points that were averaged to compute the travel time.

- variance = the variance of the data points averaged to compute the travel time.

Example:

```
3 7 2 3 900.0 2000 .15
```

The travel time between zones 3 and 7 for mode 2 in time interval 3 (5:30 am to 10 am as specified in the time intervals file) is 15 minutes (900 seconds). 2,000 data points were used to compute the travel time with a variance of .15. See also Section 8.2, Table 4.

6.2 Output Files

6.2.1 Activity Files

The following activity file is the protocol for the interaction of the TRANSIMS activity sets with the Route Planner and Traffic Microsimulator.

Format:

> The activity file consists of an ASCII file containing the activity data. Activities for a household are grouped sequentially in the activity file. Each line of the file contains tab-delimited data fields for a single activity.

Appendix A provides the meaning and format of the activity data fields. For most fields, the entry *–1* denotes an unspecified value.

The reference time is taken as 0.00 (midnight of the first day). All times are decimal numbers that denote the number of hours from 0.00. Note that each time should be given to a minimum of

- two decimal places to capture minutes, and

- four decimal places if seconds are necessary.

Each activity has a start time, end time, and duration range. The preferred time for each of these is given in terms of the two parameters of a beta distribution,

$$f(t) = C(t-L)^{a-1}(U-t)^{b-1}$$

where

- *C* is a constant,
- *L* is the lower bound of the time,
- *U* is the upper bound, and
- *a* and *b* are the parameters that specify the distribution.

The mean of the distribution is $\frac{a}{a+b}$; *a=1* and *b=1* gives a uniform distribution between *L* and *U*, and the larger *a* and *b* are, the more peaked the distribution.

6.2.2 Problem File

The Activity Generator writes entries to a problem file when certain conditions are encountered during activity generation for a household. These conditions include

- insufficient vehicles in the household for the driving trips,
- failure to find a suitable driver for intra-household shared rides,
- incomplete match between the survey household and the synthetic household, and
- shared ride participants with different departure or destination locations.

The name of the problem file is specified with a configuration file key—`ACT_PROBLEM_FILE`.

Format:

> The problem file is composed of entries, one per line, that contain an integer denoting a problem type, the number of optional fields to follow, and information specific to the type of problem. The problem types are described in

Table 4.

Table 4. Activity Generator problem types.

Problem Type	Description
1	Failed to find driver for shared ride.
2	Failed to match synthetic household with survey household.
3	Arrived too late at this activity.
4	Failure to adjust times for shared ride activity.
5	Unable to assign a vehicle for a driving trip.
6	Incomplete match between survey household and synthetic household.
7	Activity has underage driver, where driving age is defined as equal to or older than the age specified by the configuration file key `ACT_ADULT_MINIMUM_AGE.`
8	Activity is school activity and person's age is not within school age ranges specified by configuration file keys.
9	Shared ride participants do not have a common departure location.
10	Shared ride participants do not have a common destination location.
11	Vehicle location is different from departure location of first driving trip.

The problem types are defined as an enumeration (`ActivityProblemType`) in *ACT/ActivityProblem.h*.

Format for types 1 and 5 is

```
   <type> 3 <HH Id> <Person Id> <Activity Id>
```

Format for types 2 and 6 is

```
   <type> 1 <HH Id>
```

Format for types 3 and 4 is

```
   <type> 5 <HH Id> <Person Id> <Activity Id> <Start Time> <End Time>
```

Format for types 7, 8, 9, and 10 is

```
   <type> 4 <HH Id> <Person Id> <Activity Id> <integer value>
```

Integer value for types 7 and 8 is the age of the traveler. Integer value for types 9 and 10 is the location of the activity.

Format for type 11 is

```
   <type> 5 <HH Id> <Person Id> <Activity Id> <destination location> <departure location>
```

6.3 Library Files

Activity library files are shown in Table 5.

Table 5. Activity library files.

Type	Filename	Description
Binary Files	*libTIO.a*	The TRANSIMS Interfaces library.
Source Files	*actio.h*	The activity data structures and interface functions.
	activityio.c	The activity interface functions source file.

6.4 Configuration File Keys

Appendix B lists the TRANSIMS configuration file keys that specify the location of input and output files and required parameters to run the Activity Generator.

Appendix C lists the TRANSIMS configuration file keys that specify the location of input and output files and required parameters to run the Activity Regenerator.

7. Population Converter

The Activity Generator requires that the synthetic population have demographics that exactly match the variables in the regression tree. The population converter program, *PopConverter*, converts a located TRANSIMS synthetic population with demographics derived from the census into a population with demographics that exactly match the variables from the Portland survey. The households in the TRANSIMS population should have the following optional demographics from the 1990 census data: `R18UNDR, RWRKR89,` and `RHHINC`. Each person in the household should have the following demographics from census data: `AGE, RELAT1, SEX,` and `WORK89`.

The household demographics in the regression tree derived from the Portland survey are:

- household size,
- income,
- age less than 5,
- age 5 to 15,
- age 26 to 45,
- household age,
- number of workers, and
- household density.

The population converter produces the demographics in the regression tree from the census demographics in the TRANSIMS population. The household density variable is determined from land-use data at the located household's home activity location. The land-use data for activity locations is part of the TRANSIMS network activity location table. The person demographics from the census data are used, but the demographic names are changed to `RELATE, WORK, GENDER,` and `AGE`.

The population converter program must be changed if other demographic variables are used in the regression tree or if different demographics from the census data are in the synthetic population.

Appendix D lists the TRANSIMS configuration file keys that specify the location of input and output files and required parameters to run the Population Converter.

7.1 Usage

The population converter uses three configuration file keys from a TRANSIMS configuration file.

1) `ACT_HHDENSITY_HEADER` is used to specify the column header in the activity location table that contains the household density data.

2) `POP_LOCATED_FILE` is used to specify the name of the file containing the located TRANSIMS population.

3) `ACT_POPULATION_FILE` is the name of the file where the converted population will be written.

The population converter has one command line argument, which is the name of the TRANSIMS configuration file.

```
% $TRANSIMS_HOME/bin/PopConverter   <configuration filename>
```

8. Trip Table Activity Generator

The Trip Table Activity Generator produces activities from the entries in a trip table. The trip table contains information about the number of trips between zones. A time table that contains the probability of a trip within a time range is also required..

The Trip Table Activity Generator can be used to quickly generate travelers for research purposes, as well as to produce itinerant travelers and freight trips on a transportation network. In addition to the TRANSIMS activity file, the Trip Table Activity Generator produces a population file containing single-person households for each trip and a TRANSIMS vehicle file.

Activities are generated with a car mode (`wcw` mode string). The vehicle type can be specified using the following configuration file key:

```
VEH_VEHICLE_TYPE
```

The default value of this key is `kAuto(1)`, but it can be set to produce truck vehicle types for freight trips.

The vehicle subtype can be specified by using the configuration file key `VEH_VEHICLE_SUBTYPE`.

8.1.1 Determining Origin and Destination Locations

Trip origins and destinations are chosen based on an attractor value specified for the activity locations in a zone. The locations will be chosen randomly from the activity location in the zone if no attractor is specified.

A column in the TRANSIMS network Activity Location Table identifies the zone and attractor, if specified, associated with each activity location. The column header in the Activity Location Table (defined by the configuration file key `ACT_TAZ_HEADER`) is used to determine possible origin/destination locations by matching the value in this column with the zone indexes in the trip table. The column headers, `ACT_TRIPTABLE_ORIGIN_ATTRACTOR_HEADER` and `ACT_TRIPTABLE_DESTINATION_ATTRACTOR_HEADER`, are used to determine the probability that an activity location in the zone will be chosen as the origin/destination location.

8.1.2 Determining Activity Times

Two activities are produced for each trip. The number of trips per time of day is determined from a time table that contains time ranges from 0 to 24 hours and probability of a trip within that range.

- The end time of the first activity is based on these probabilities.
- The starting time of the second activity is 3 hours plus the end of the first activity.
- The end time of the second activity is determined by the configuration file key `ACT_MAX_END_TIME`. If not set, the default value of 27.0 hours is used.

8.2 Files

8.2.1 *Trip Table* File

The *Trip Table* file contains zone pairs and the number of trips between the zones.

Format:

```
<zone 1> <zone 2> <number of trips>
```

Example:

```
1  2  340
1  3  450
```

This example specifies 340 trips from zone 1 to zone 2, and 450 trips from zone 1 to zone 3.

8.2.2 *Time Table* File

The *Time Table* file contains the lower and upper bounds of the time range in hours and the probability of a trip within the time range. The probabilities in the table should sum to 1.0. The time ranges in the table must cover a 24-hour period (0 – 24 hours).

Format:

```
<time 1>  < time 2>  <probability>
```

Example:

```
 0.0   5.0  0.1
 5 0  19.0  0.8
19 0  24.0  0.1
```

The probability of a trip between midnight (0.0) and 5:00 a.m. is 0.1, between 5:00 a.m. and 7:00 p.m. is 0.8, and between 7:00 p.m. and midnight is 0.1.

9. Zone Travel Time Generator

The Zone Travel Time Generator creates a file of mean travel times between zones by mode and time of day. This information can then be used by the Activity Generator to develop better activity times. The travel times can come from estimates by the Route Planner or actual times generated by the Traffic Microsimulator.

The travel time information is written to the travel times summary file (configuration file key `ACT_TRAVEL_TIMES_FILE`). In addition, the travel times for each leg are written to the travel times detail file, which has the same name as the summary file, with a *.all* extension.

9.1 Collator Configuration File Keys

The Collator is used to place the travel time information into the Iteration Database. The following configuration file keys must be specified in the configuration file to have the necessary information collected:

```
SEL_USE_START_ACC 1
SEL_USE_END_ACC 1
SEL_USE_DEP_TIME 1
SEL_USE_END_MODE_PREF 1
SEL_USE_MODE_STRING 1
SEL_USE_START_REGION 1
SEL_USE_END_REGION 1
SEL_UAZ_FILE 1
SEL_USE_DURATION 1
SEL_USE_T_TOTAL 1
```

Note that `SEL_USE_T_TOTAL` is only necessary to use travel times generated by the Traffic Microsimulator. If both the Route Planner and the Traffic Microsimulator data exist for a particular traveler, the Traffic Microsimulator data are used.

9.2 Travel Time Summary File

The travel time summary file contains information about the mean travel time for travel between zones for a particular travel mode and time of day. Activity locations for the start and end of a trip are mapped into zones using the information in the network activity table. The configuration file key `ACT_TAZ_HEADER` specifies the column that contains the zone for each activity location. The fields of the travel time summary file are described in Table 5.

Table 5. Travel time summary file fields.

Field Name	Description
`FZONE`	Zone of the starting activity location.
`TZONE`	Zone of the ending activity location.
`MODE`	Travel mode.
`INTERVAL`	Time interval during which the travel started.
`DURATION`	Mean duration of the trip, in seconds.
`COUNT`	Number of trips used to compute the duration.
`VARIANCE`	Variance in the durations for this trip.

9.3 Travel Time Detail File

The travel time detail file contains information about the individual trips that are summarized in the travel time summary file. In addition to the zones used in the summary file, arbitrary polygons can be specified as User Analysis Zones (UAZ). More information about UAZs can be found in Chapter Six (*Selectors/Iteration Databases*). the fields of the travel time detail file are described in Table 6.

Table 6. Travel time detail file fields.

Field Name	Description
`TravelerId`	The ID of the traveler for this trip.
`FromAL`	Starting activity location.
`FromZone`	Zone of the starting activity location.
`FromXZone`	UAZ of the starting activity location.
`ToAL`	Ending activity location.
`ToZone`	Zone of the ending activity location.
`ToXZone`	UAZ of the ending activity location.
`Mode`	Travel mode.
`Time`	Starting time of the trip in seconds.
`Duration`	Duration of the trip, in seconds.

9.4 Time Intervals

The time intervals to use are read by the file specified by the `ACT_TRAVEL_TIME_INTERVALS_FILE` configuration file key. Each line of the interval file consists of a start time, a stop time, and an interval ID. A time that is greater than or equal to the start time and less than the stop time is considered to be in the interval given by ID. All times are given in seconds. If a time falls in more than one interval, the interval that occurs earliest in the interval file is used. If a time does not fall into any interval, it is placed in interval 0.

For example, the interval file:

```
BEGIN  END    INTERVAL
25200  32400  1
32400  57600  2
57600  68400  1
```

describes three intervals:

```
interval 0 0:00 to 7:00 and 18:00 to 24:00
interval 1 7:00 to 9:00 and 16:00 to 18:00
interval 2 9:00 to 16:00
```

9.5 Modes

Modes are read from the file specified by the ROUTER_MODE_MAP_FILE configuration file key.

Travel times for bike mode (mode string wiw) are converted to walk times and included with the walk mode travel times. This conversion is done using the speeds specified by the configuration file keys ROUTER_WALKING_SPEED and ROUTER_BIKING_SPEED.

10. Survey Activity Test Program

The survey activity testing program, *SurveyActTester*, executes the location choice, shared-ride assignment, and vehicle assignment algorithms in the Activity Generator on the survey households. The process and resulting activity set will show inconsistencies in the survey household activity patterns that must be corrected before overlaying the patterns on a synthetic population.

Survey households that are located on the transportation network are required. In addition, a fake census tract and block group should also be assigned to the survey households. One method to obtain a population composed of survey households is to copy the locations, tract, and block group from a located synthetic population to the synthetic households and create a TRANSIMS population file containing the survey households with the information for location, tract, and block group from the synthetic household. The demographics and other household and person information from the survey households is preserved in the new file. The locations, tract, and block group in the survey population may not be realistic but will serve for testing purposes. This new file containing the survey households located on the network is specified using the configuration file key `ACT_POPULATION_FILE` in the configuration file used by the survey activity testing program.

The Vehicle Generator program (*Vehgen*) can be executed using this new survey population file to generate a TRANSIMS vehicle file for the survey households. The survey population file should be specified as the `POP_LOCATED_FILE` configuration file key in the configuration file used by the Vehicle Generator. The vehicle file is necessary for the vehicle assignment algorithm in the Activity Generator.

The result will be a set of TRANSIMS activities for the survey households. The Activity Generator problem file and log file will contain warning and error messages for any survey household with a problematic activity list. The output of the *SurveyActTester* program should be redirected into the log file.

10.1 Usage

SurveyActTester uses the same configuration file keys as the Activity Generator, see Appendix B.

The survey activity testing program has one command line argument, which is the name of a TRANSIMS configuration file.

```
% $TRANSIMS_HOME/bin/SurveyActTester <configuration filename> > <log-file name>
```

Appendix A: Activity Data Definitions and Format

Field	Description	Allowed Values
`Household ID`	Each household has a unique household ID. Each Group Quarters is given one household ID. These numbers are assigned in the population file.	integer
`PersonID`	Each person is given a unique ID in the population file.	integer
`ActivityID`	Each activity in the household has a unique ID.	integer >0
`Activity Type`	The definition of activity types may vary. Meaning of the integer value must be specified for each activity set.	integer: 0 through n: Example: 0 = Home 1 = Work 2 = Shop 3 = Visit 4 = Social/Rec 5 = Other 6 = Serve Passenger 7 = School 8 = College
`Activity Priority`	Priority that indicates the importance of the activity. Higher values mean that the activity may be skipped. Lower values mean that the activity must be done.	integer: 0 – 9
`Starting Time Lower Bound`	Earliest time the activity can start.	decimal
`Starting Time Upper Bound`	Latest time an activity can start.	decimal
`Preferred Starting Time` *`a`* `parameter`	The time the Route Planner will use as the best guess for the starting time. If this number is –1, the average of the upper and lower bounds is used.	decimal
`Preferred Starting Time` *`b`* `parameter`	The time the Route Planner will use as the best guess for the starting time. If this number is –1, the average of the upper and lower bounds is used.	decimal
`Ending Time Lower Bound`	The earliest time the activity can end.	decimal
`Ending Time Upper Bound`	The latest time the activity can end.	decimal
`Preferred Ending Time` *`a`* `parameter`	The time the Route Planner will use as the best guess for the activity ending time. If this number is –1, the average of the lower and upper bounds is used.	decimal
`Preferred Ending Time` *`b`* `parameter`	The time the Route Planner will use as the best guess for the activity ending time. If this number is –1, the average of the lower and upper bounds is used.	decimal
`Duration Lower Bound`	The shortest length of the activity.	decimal
`Duration Upper Bound`	The longest length of the activity.	decimal
`Duration a parameter`	The Route Planner will use this time as the best guess of the activity duration. If this number is –1, the average of the upper and lower bound is used.	decimal

Field	Description	Allowed Values
`Duration b parameter`	The Route Planner will use this time as the best guess of the activity duration. If this number is –1, the average of the upper and lower bound is used.	decimal
`Mode Preference for Arriving at the Activity`	This number represents a grammar string that defines the mode preference to the Route Planner (wcw, wtw, …). The correspondence between integer values and possible grammar strings is contained in an external file.	integer
`Vehicle ID`	The vehicle ID for all activities with a mode preference of `private auto`, either as driver or as passenger. This field should be set to –1 for all other modes.	integer
`Number of Possible Locations for Activity`	The number of possible locations in the `List of Locations` field if value is 1 or greater. The value 0 is not allowed. If this field is –1, the single value in the `List of Locations` field is an index into a group of activities.	-1, integer ≥ 1
`List of Activity Locations`	If the `Number of Possible Locations` field is 1 or greater, this field contains a list of activity location IDs where an activity may take place. If the `Number of Possible Locations` field is –1, this field contains a number that is an index into a group of activities.	integer [integer] …
`Number of Other Participants in the Activity`	The number of others in the population who might participate and use the same transportation (e.g., the same car). The number is 0 if the person is to travel alone to the activity.	integer
`List of Other Participants`	The person IDs of participants using the same transportation. This field should be present only when the value of the `Number of Other Participants` field is > 0. The first entry in this list is always the person ID of the driver. For passengers, the driver ID is the only entry in this list. For drivers, the person IDs of the passengers are appended to the first entry in the list, the driver ID.	[integer] [integer] …

Field	Description	Allowed Values
`Activity Time Priority`	A priority assigned to the activity that indicates whether start time, end time, and/or duration of the activity may be changed by the other modules such as the Route Planner. Allowed Values: 0 = start, stop, and duration of activity may change. 1 = activity must start at the specified start time. 2 = activity must stop at the specified end time. 3 = activity must start and end at the specified times. 4 = duration of the activity must not be changed. 5 = activity must start at the specified start time and last for the specified duration. 6 = activity must last for the specified duration and stop at the specified end time. 7 = activity must start and stop at the specified times and last for the specified duration.	integer: 0 – 7 (see value definitions in box at the left)
`Notes`	The character string used for annotations; free format (may be blank).	255 characters

Appendix B: Activity Generator Configuration File Keys

Configuration File Key	Description
ACT_ACCESS_HEADER	The user data column header in the network activity location file used to specify access to transit.
ACT_ACTIVITY_TYPE	The activity types used by the Activity Generator. The base key must be followed with _n where n is an integer to indicate the n^th specification of the activity type (non-negative integer).
ACT_ADJUST_ACTIVITY_TIMES	A control for adjusting survey activity times by considering the travel time between activities. Integer values 0 and 1: 0 = no adjustments 1 = adjust for travel time Default = 1
ACT_ANCHOR_ACTIVITY_TYPE	An activity type that will be considered an anchor activity when determining the locations on a traveler's tour. The base key must be followed by _n where n is an integer to indicate the n^th specification of anchor activity type. If no anchor activity types are specified, home, work, and school types are used as anchor activities.
ACT_AUTOMOBILE_MODE*	A mode that will be considered a personal automobile mode by the Activity Generator. Automobile modes are used to determine shared rides and vehicle assignments. At least one mode must be specified. Park-and-ride modes can be specified as automobile modes in the Activity Generator.
ACT_BICYCLE_MODE	The number of the bicycle mode (wiw) (integer).
ACT_BLOCKGROUP_HEADER	The user data column header in the network activity location file used to specify block group.
ACT_DECISION_TREE_FILE	The name of the file containing the regression tree for the Activity Generator.
ACT_DEFAULT_CAR_SPEED	The default speed for automobiles in meters/second (floating-point number). Default = 15.0
ACT_DEFAULT_INTRAZONE_DISTANCE	The average distance, in meters, of a trip within a zone.
ACT_DEFAULT_TRANSIT_MODE	The number of the default transit mode (wtw) (integer).
ACT_DEFAULT_TRANSIT_SPEED	The default transit speed in meters/second (floating-point number). Default = 10.0

Configuration File Key	Description
ACT_END_OF_DAY_TIME_RANGE	The time range in hours for lower and upper bounds of start and end times for the end-of-day activity (positive floating-point number). Default = 0.75
ACT_HOME_ACTIVITY_TYPE	The number of the home activity type (non-negative integer).
ACT_HOME_DURING_DAY_TIME_RANGE	The time range in hours for lower and upper bounds of start and end times for non-work activities originating at home (positive floating-point number). Default = 0.75
ACT_HOME_HEADER	The user data column header in the network activity location file used to specify single family home locations.
ACT_HOUSEHOLD_FILE	The name of a file containing a list of household IDs from the synthetic population for which activities will be generated. Activities will be generated only for those households on the list. This key is optional and if not specified, activities will be generated for all households in the population (ACT_POPULATION_FILE).
ACT_INITIAL_HOME_TIME_RANGE	The time range in hours for lower and upper bounds of start and end times for the initial at-home activity (positive floating-point number). Default = 0.75
ACT_LOCATION_CHOICE_EXPONENT	The power to which the exponential function in the location choice algorithm will be raised. Floating-point value. Default – 1.0
ACT_LOCATION_HEADER	The header for the activity type for activity locations in the network activity location table. The headers must correspond to the activity types defined with the ACT_ACTIVITY_TYPE_n keys. The base key must be followed with _n where n is an integer to indicate the nth specification of the header.
ACT_LOG_FILE	The name of the logfile for the Activity Generator and Regenerator. Default = *ActivityGenerator.log*
ACT_MAGIC_MOVE_MODE	The number of the magic move mode (wkw) (integer).
ACT_MAX_END_TIME*	The maximum end time for an activity in hours past midnight on the starting day (positive floating-point number). Default = 24.0
ACT_MINIMUM_ADULT_AGE	The minimum age for an adult in years. Persons younger than this value will be considered children when matching synthetic households with survey households in the activity generator. Default = 16

Configuration File Key	Description
ACT_MODE_WEIGHT_FILE	The name of the file containing mode coefficients for the activity types. This must contain a coefficient for every mode and activity type.
ACT_MULTI_FAMILY_HEADER	The user data column header in the network activity location file used to specify multifamily home locations.
ACT_OUT_OF_HOME_TIME_RANGE	The time range in hours for lower and upper bounds of start and end times for non-work activities that do not originate at home (positive floating-point number). Default = 0.75
ACT_PERSON_DEMOG_AGE_HEADER	The name of the age demographic header for the persons in the population used by the Activity Generator.
ACT_PERSON_DEMOG_GENDER_HEADER	The name of the gender demographic header for the persons in the population used by the Activity Generator.
ACT_PERSON_DEMOG_RELATION_HEADER	The name of the relationship demographic header for the persons in the population used by the Activity Generator.
ACT_PERSON_DEMOG_WORKER_HEADER	The name of the worker demographic header for the persons in the population used by the Activity Generator.
ACT_POPULATION_FILE	The name of the file containing a located synthetic population with household and person demographics that exactly match the variables in the Activity Generator regression tree. This file is output from the population converter program.
ACT_PRIORITY*	The priority for the activity type. The priorities must correspond to the activity types defined with the ACT_ACTIVITY_TYPE_n configuration file keys. The base key must be followed with _n where n is an integer to indicate the nth specification of the priority. Defaults = work activity type 2 school activity type 3 home activity type 1 all other activity types 7
ACT_PROBLEM_FILE	The name of the file where information about problems that occurred during activity generation will be written. Default = *act.problems*
ACT_RANDOM_SEED	The random number seed used by activity generators.

Configuration File Key	Description
ACT_REQUIRED_HH_DEMOG	The required household demographics in the synthetic population used by the Activity Generator. The base key must be followed with _n where n is an integer to indicate the n^th^ specification of required demographics. The demographics must exactly match and be ordered the same (1 - n) as the demographic variables in the Activity Generator's regression tree.
ACT_SCHOOL_ACTIVITY_TYPE	The number of the school activity type (non-negative integer).
ACT_SCHOOL_LOCATION_ATTRACTOR_VALUE	The value of the activity location attractor for the school activity type. The base key must be followed by _n where n is an integer to indicate the n^th^ specification of the activity location attractor value. The specifications should be linked to the school age ranges specified. If no school activity location attractor values are specified, the activity location attractors for the school activity type are used as weights.
ACT_SCHOOL_LOWER_BOUND	The lower bound of a school age range. The base key must be followed by _n where n is an integer to indicate the n^th^ specification of the school age range lower bound.
ACT_SCHOOL_UPPER_BOUND	The upper bound of a school age range. The base key must be followed by _n where n is an integer to indicate the n^th^ specification of the school age range upper bound.
ACT_SCHOOL_ZONE_ATTRACTOR_VALUE	The value of the zone attractor for the school activity type. The base key must be followed by _n where n is an integer to indicate the n^th^ specification of the zone attractor value. The specifications should be linked to the school age ranges specified. If no school zone attractor values are specified, the zone attractors for the school activity type are used as weights.
ACT_SHARED_RIDE_DISTANCE_RANGE*	The distance range in meters for matching activity locations for shared rides. Default = 10
ACT_SHARED_RIDE_TIME_RANGE_MAX*	The maximum time range in minutes for matching activities for shared rides. Default = 1
ACT_SHARED_RIDE_TIME_RANGE_MEDIUM	Medium time range in minutes for matching shared rides. Default = 1
ACT_SHARED_RIDE_TIME_RANGE_MIN*	The minimum time range in minutes for matching activities for shared rides. Default = 1

Configuration File Key	Description
ACT_SURVEY_ACTIVITY_FILE	The name of the file containing activity patterns for the survey households.
ACT_SURVEY_HOUSEHOLD_FILE	The name of the file containing the survey household population and demographics.
ACT_SURVEY_WEIGHTS_FILE	The name of the file containing the relative weights of the survey households.
ACT_TAZ_HEADER	The user data column header in the network activity location file used to specify traffic analysis zone.
ACT_TIME_PRIORITY*	Specifies time priority for the activity type. The time priorities must correspond to the activity types defined with the ACT_ACTIVITY_TYPE_n configuration file keys. The base key must be followed with _n where n is an integer to indicate the n^{th} specification of the time priority. The allowed values of the time priority configuration file keys are described in Appendix A (*Field Activity Time Priority*). Defaults = work activity type 3 school activity type 3 home activity type 0 all other activity types 0
ACT_TRACT_HEADER	The user data column header in the network activity location file used to specify census tract.
ACT_TRAVEL_TIMES_FILE	The name of the file containing travel time information between zones.
ACT_TRAVEL_TIME_FILE_MODES	Specifies the modes where the travel time will be determined from the travel times file. Modes are specified as a semicolon-separated list of integer mode values. This key is optional and if not specified and a travel time file is specified (ACT_TRAVEL_TIMES_FILE), the Activity Generator will look for all modes in the travel times file. Example: ACT_TRAVEL_TIME_FILE_MODES 1;3;7
ACT_TRAVEL_TIME_INTERVALS_FILE	The file containing travel time interval information.
ACT_USER_FUNCTION_MODES	Specifies the modes for which the user function is valid. Modes are specified as a semicolon-separated list of integer mode values. Example: ACT_USER_FUNCTION_MODES 1;2;9

Configuration File Key	Description
ACT_USER_FUNCTION_PARAMETERS	Specifies the optional parameters to the user function. Parameters are specified as a semicolon-separated list of floating-point values. If specified, the floating-point values are passed into the user function as a vector of doubles. Example: `ACT_USER_FUNCTION_PARAMETERS 0.2;4.1;-.00025`
ACT_USER_FUNCTION_STRING_PARAMETERS	Specifies the optional user-defined string parameters to the user function. Parameters are specified as a semicolon-separated list of string values. If specified, the string values are passed into the user function as a vector of strings. String values must not contain whitespace. Example: ACT_USER_FUNCTION_STRING_PARAMETERS /usr/transims/data/data.1; transit_only;bridge_crossing
ACT_USER_FUNCTION_SHARED_OBJECT	Specifies the full path name of the shared object that contains the user function. If specified, the user function object is loaded at runtime and will be used in the location choice models for all modes specified by the `ACT_USER_FUNCTION_MODES` configuration file key. Example: ACT_USER_FUNCTION_SHARED_OBJECT /usr/transims/work/UserZoneFunction.o
ACT_USER_FUNCTION_NAME	Specifies the name of the user-defined function in the shared object specified by `ACT_USER_FUNCTION_SHARED_OBJECT.` Default value = *UserZoneFunction*
ACT_WALKING_MODE	The number of the walking mode (`w`) (integer).
ACT_WORK_ACTIVITY_TYPE	The number of the work activity type (non-negative integer).
ACT_WORK_HEADER	The user data column header in the network activity location file used to specify work locations.
ACT_WORK_TIME_RANGE	The time range in hours for lower and upper bounds of start and end times for work activities (positive floating-point number). Default = 0.25
ACT_ZONE_HEADER	Used to specify the header for the zone attractors, which must match the activity types (`ACT_ACTIVITY_TYPE_n`). The base key must be followed with `_n` where `n` is an integer to indicate the `n`th specification of the header.
ACT_ZONE_INFO_FILE	The name of the file containing zone attractor data by activity type for the Activity Generator.
ACTIVITY_FILE	The name of the TRANSIMS activity file for the household.
NET_ACTIVITY_LOCATION_TABLE	The network activity location table name.
NET_DIRECTORY	The directory where the network files reside.

Configuration File Key	Description
NET_LINK_TABLE	The network link table name.
NET_NODE_TABLE	The network node table name.
ROUTER_BIKING_SPEED	The approximate speed in meters/second for bicycles.
ROUTER_WALKING_SPEED	The approximate speed in meters/second for walking trips.
VEHICLE_FILE	The name of the TRANSIMS vehicle file for the population.

* These keys are optional.

Appendix C: Activity Regenerator Configuration File Keys

Configuration File Key	Description
`ACT_FEEDBACK_FILE`	The file containing a list of travelers and associated commands for activity regeneration.
`ACT_PARTIAL_OUTPUT`	The file that will be output from partial regeneration of activities.

Appendix D: Population Converter Configuration File Keys

Configuration File Key	Description
`ACT_HHDENSITY_HEADER`	The column header of the household density values in the network activity location tables.
`ACT_POPULATION_FILE`	The file containing a located synthetic population with household and person demographics that exactly match the variables in the Activity Generator regression tree. This file is output from the Population Converter program.
`POP_LOCATED_FILE`	The file containing the located population.

Appendix E: Trip Table Activity Generator Configuration File Keys

Configuration File Key	Description
ACT_HOME_ACTIVITY_TYPE	The number of the home activity type. Non-negative integer. Default = 1
ACT_MAX_END_TIME*	The maximum end time for an activity in hours past midnight on the starting day (positive floating-point number). Default = 24.0
ACT_RANDOM_SEED	The seed for the random number system.
ACT_TAZ_HEADER*	The column header in the network activity location file that contains the zone information. Default = TAZ
ACT_TRIP_TABLE_OUTPUT*	The name of the activity file that will be output from the Trip Table Activity Generator.
ACT_TRIP_TABLE_VEHICLE_FILE	The name of the vehicle file that will be output from the Trip Table Activity Generator.
ACT_TRIPTABLE_DESTINATION_ATTRACTOR_HEADER	The column header in the network activity location file for the location attractor for the origin of trips generated using trip tables.
ACT_TRIPTABLE_FILE	The name of the file containing the trip table matrix.
ACT_TRIPTABLE_ORIGIN_ATTRACTOR_HEADER	The column header in the network activity location file for the location attractor for the origin of trips generated using trip tables.
ACT_TRIPTABLE_STARTING_HH_ID*	The starting household ID for households generated from trip table matrices. Default = 1
ACT_TRIPTABLE_STARTING_PERSON_ID*	The starting person ID for travelers generated from trip table matrices. Default = 1
ACT_TRIPTABLE_STARTING_VEHICLE_ID*	The starting vehicle ID for vehicles generator from trip table matrices. Default = 1
ACT_TRIPTIME_FILE	The name of the file containing the time of day trip table data.
ACT_WORK_ACTIVITY_TYPE*	The number of the work activity type. Non-negative integer. Default = 2
MODE_MAP_FILE	The file containing mapping between mode strings and integer values. The string "*wcw*" must be in this file.
NET_ACTIVITY_LOCATION_TABLE	The network activity location table. Must contain a column that has the zone number for the activity locations.
NET_DIRECTORY	The directory where the network tables reside.
NET_LINK_TABLE	The network link table.
NET_NODE_TABLE	The network node table.
NET_PARKING_TABLE	The network parking table.
NET_PROCESS_LINK_TABLE	The network process link table.
NET_TRANSIT_STOP_TABLE	The network transit stop table (may be an empty table).

Configuration File Key	Description
POP_TRIPTABLE_FILE	The name of the population file that will be output from the trip table activity generator.
VEH_VEHICLE_SUBTYPE*	The subtype of the vehicle fleet will be generated. Default = 0
VEH_VEHICLE_TYPE*	The type of vehicles that will be generated. Default value is assigned from a type enumeration in the TRANSIMS Network = 1 (kAuto).

* Optional configuration file keys. If not specified, will use default values.

Appendix F: Activity Generator, Activity Regenerator Error Codes

Error codes for the Activity Generator, Activity Regenerator are in the range 11000 – 11999.

Table 7. Activity Generator, Activity Regenerator error codes.

Code	Description
11001	Caught signal.
11002	Assertion failed.
11003	Network exception occurred.
11004	Standard exception occurred.
11005	Unknown exception occurred.
11006	Invalid program usage.
11007	Memory allocation failed.
11008	Failed to open file for reading.
11009	Failed to open file for writing.
11010	Invalid array index.
11011	Invalid file descriptor.
11012	Failed to read record from file.
11013	Activity type not specified.
11014	Zone header not specified.
11015	Demographic header not specified.
11016	Activity location user data header not specified.
11017	Configuration key for file not specified.
11018	Mode not specified.
11019	Network directory not specified.
11020	Network activity location table not specified.
11021	Invalid zone number
11022	Specified household demographic not found in population.
11028	Specified person demographic not found in population.
11024	Time array size is too small.
11025	No driver activity associated with shared ride activity.
11026	Driver activity has invalid location.
11027	Failed to read record from file.
11028	Failed to read header from file.
11029	Invalid population file header.
11030	Invalid activity location table header.
11031	Invalid header in zone attractor file.
11032	Failed to construct network.
11033	No coefficient for given mode.
11034	Failed to choose anchor location for non-anchor activity.
11035	Failed to choose school for non-school activity.
11036	Home location not in home zone.

Code	Description
11037	Regression tree does not exist.
11038	Terminal node in regression tree is empty.
11039	Invalid node number in regression tree.
11040	Failed to find matching survey household for synthetic household.
11041	Survey household has no persons.
11042	Failed to find specified person in survey household.
11043	Failed to choose location for activity.
11044	Activity has invalid location.
11045	Invalid activity location ID.
11046	Arithmetic overflow…exp function is out of range.
11047	Last activity for person is not at end of trips.
11048	Number of specified school age ranges is inconsistent with specified zone/location values.

Appendix G: Trip Table Activity Generator Error Codes

Error codes for the Trip Table Activity Generator are in the range 12000 – 12999.

Table 8. Trip Table Activity Generator error codes.

Code	Description
12001	Caught signal.
12002	Assertion failed.
12003	Network exception has occurred.
12004	Exception has occurred.
12005	Unknown exception has occurred.
12006	Memory allocation failed.
12007	Invalid program usage.
12008	Failed to open file for reading.
12009	Failed to open file for writing.
12010	Failed to read record from file.
12011	Mandatory file not specified.
12012	Mode not found in mode file.
12013	Failed to construct network.
12014	No user data for specified header in network activity location table.
12015	Failed to find location for activity.
12016	Failed to create index for file.

Chapter Three: Index